

1 No hay escena de la prehistoria absolutamente tan  
2 viva como la de las luchas mortales de grandes  
3 bestias en los hoyos de alquitrán. Con el ojo de la  
4 mente uno ve los dinosaurios, mamuts, y los tigres  
5 dientes de sable que luchan contra el apretón del  
6 alquitrán. Cuanto más feroz es la lucha, más se  
7 enredan en el alquitrán, y no hay bestia tan fuerte o  
8 tan experta por eso en última instancia se hunden.

9

10 La programación de grandes sistemas, en la década  
11 pasada han sido como un pozo de alquitrán, grandes y  
12 poderosa bestias cayeron violentamente dentro de él.

13

14 La mayoría han emergido con sistemas funcionando,  
15 pocos han cumplido metas, horarios y presupuestos.  
16 Grandes y pequeños, masivos o wiry (PARTICULARES ¿?),  
17 un equipo luego de otro se ha enredado en el  
18 alquitrán. Ninguna cosa parece ser la dificultad,  
19 cualquier pata en particular puede ser retirada. Pero  
20 la acumulación de factores simultáneos que  
21 interactúan entre si traen lento y más lento  
22 movimiento.

23 Cada uno parece haber sido sorprendido por la  
24 viscosidad (PEGAJOSO??) del problema, y es duro  
25 discernir la naturaleza de él. Pero debemos intentar  
26 entenderlo si debemos solucionarlo.

27 Por lo tanto permitámonos comenzar por identificar el  
28 arte de la programación de sistemas y las alegrías y  
29 penurias inherentes a este.

30

31 EL PRODUCTO PROGRAMACION DE SISTEMAS.

32 De vez en cuando uno lee en el periódico crónicas de  
33 cómo dos programadores en un garaje remodelado han  
34 construido un programa importante que sobrepasa los  
35 mejores esfuerzos de grandes equipos. Y cada  
36 programador está preparado para creer tales cuentos,  
37 porque él sabe que él podría construir cualquier  
38 programa mucho más rápidamente que las 1000 líneas de  
39 código/año divulgado para los grandes equipos.

40 ¿Por qué entonces no han sido reemplazados todos los  
41 equipos industriales por "dúos de garaje"?, uno tiene  
42 que mirar hacia lo que se está produciendo.

43

44 En la parte superior derecha de la figura 1.1,  
45 aparece un PROGRAMA, Es completo en sí mismo, listo  
46 ser puesto en funcionamiento por el autor en el sistema  
47 en el cual fue desarrollado. Eso es lo producido  
48 comúnmente en garajes, y ése es el objeto que el  
49 programador individual utiliza en la estimación de la  
50 productividad.

51

52 Figura 1.1 Evolución del producto de los sistemas de  
53 programación

54

55 Hay dos maneras en las que un programa puede  
56 convertirse en un objeto más útil, pero más costoso.  
57 Estas dos maneras están indicadas en los límites del  
58 diagrama.

59

60 Moviéndose abajo a través del límite horizontal, un  
61 programa se convierte en un producto de programación.  
62 Éste es un programa que se puede funcionar, probar,  
63 reparar, y ampliar por cualquiera. Es usable en  
64 muchos ambientes de funcionamiento, para muchos

65 sistemas de datos. Para convertirse en un producto de  
66 programación generalmente usable, un programa se debe  
67 escribir de una manera generalizada. Particularmente  
68 el rango y la forma de las entradas se deben  
69 generalizar tanto como el algoritmo básico  
70 razonablemente lo permita. A continuación, el  
71 programa debe ser probado exhaustivamente, *so that it*  
72 *can be depended upon*. Esto significa que un banco  
73 substancial de casos de prueba, explorando el rango  
74 de entradas y probando sus límites, debe ser  
75 preparado, puesto en funcionamiento y registrado.  
76 Finalmente, llevar un programa a un producto de  
77 programación requiere su documentación cuidadosa, de  
78 modo que cualquier persona pueda utilizarla,  
79 corregirla, y extenderla. En general, estimo que un  
80 producto de programación cuesta por lo menos tres  
81 veces más que un programa depurado que realice la  
82 misma función.  
83

84 Moviéndose a través del límite vertical, un programa  
85 se convierte en un componente en un sistema de  
86 programación. Ésto es una colección de programas que  
87 obran recíprocamente, coordinados en funcionamiento y  
88 disciplinados en formato, de modo que el ensamblaje  
89 constituya una facilidad entera para tareas grandes.  
90 Para convertirse en un componente del sistema de  
91 programación, un programa debe ser escrito de modo  
92 que cada entrada y salida se defina en sintaxis y en  
93 la semántica con interfaces precisamente definidas.  
94 El programa debe también ser diseñado de modo que  
95 utilice solamente un presupuesto estimado de  
96 recursos, espacio de memoria, dispositivos de  
97 entrada-salida, tiempo de computadora. Finalmente el  
98 programa se debe probar con otros componentes del  
99 sistema, en todas las combinaciones previstas. Estas  
100 pruebas deben ser extensas porque el número de casos  
101 crece de manera combinatoria.  
102  
103

104 Se desperdicia tiempo, porque los errores sutiles se  
105 presentan en interacciones inesperadas de componentes  
106 que han sido depurados.

107 Un componente en un sistema de programación cuesta al  
108 menos 3 veces que un programa "Stand-alone" que  
109 realice las mismas funciones. El sistema será más  
110 costoso entre mas componentes incluya.

111

112 En la esquina inferior derecha se encuentran los  
113 sistemas productos de programación. Estos se  
114 diferencian de un programa simple, en todo lo antes  
115 dicho. Tiene un costo nueve veces mayor, pero es el  
116 objeto verdaderamente útil, el producto previsto por  
117 la mayoría de los esfuerzos de programación de  
118 sistemas.

119

120

121 LAS ALEGRIAS DEL ARTE.

122 ¿Por qué es la programación divertida? ¿Qué placeres  
123 puede su practicante recibir como recompensa?.

124

125 Primero está la alegría escarpada de hacer cosas.  
126 Como el niño se deleita con su torta de barro, el  
127 adulto disfruta construyendo cosas, especialmente  
128 cosas que el mismo diseña. Yo pienso que esta delicia  
129 tiene que ser una imagen del placer de Dios haciendo  
130 las cosas, un placer visto en la distinción de cada  
131 hoja y cada copo de nieve.

132

133 Segundo, es el placer de hacer cosas que son útiles a  
134 otras personas. En nuestro interior queremos que  
135 otros utilicen nuestro trabajo y lo encuentren útil.

136

137 A este respecto el sistema de programación no es  
138 esencialmente diferente del primer portalápiz de  
139 arcilla hecho por el niño "para la oficina del papá."

140

141

142

143 Tercero es la fascinación de configurar complejos  
144 objetos como rompecabezas de interconectar partes  
145 móviles y viéndolas trabajar en sutiles ciclos,  
146 jugando fuera de las consecuencias de principios  
147 contruidos desde el inicio. El computador programado  
148 tiene toda la fascinación de la maquina de "pinball"  
149 o del mecanismo de jukebox llevado al máximo.

150

151 El cuarto es la alegría de siempre aprender, que  
152 despegas de la naturaleza no repetitiva de la tarea.  
153 De una u otra forma el problema es siempre nuevo y de  
154 su solución se aprende algo nuevo: algunas veces  
155 teórico, algunas veces algo práctico, y algunas veces  
156 ambos.

157

158 Finalmente, hay el placer del trabajo en un medio tan  
159 manejable. El programador, como el poeta, trabaja  
160 ligeramente retirado de la naturaleza pura de las  
161 cosas. El construye sus castillos en el aire, del  
162 aire, creando por un esfuerzo de la imaginación.  
163 Pocos medios de creación son tan flexibles, fáciles  
164 de pulir y revisar, fáciles para la creación de  
165 grandes estructuras conceptuales. (Como podremos ver  
166 mas tarde esto es muy tractability has its nuestros  
167 problemas.)

168

169 Sin embargo la construcción del programa, a  
170 diferencia de las palabras del poeta, es real en el  
171 sentido de que se mueve y trabaja, que produce  
172 resultados visibles aparte de la construcción de sí  
173 mismo. El imprime resultados, dibuja figuras, produce  
174 sonidos, mueve brazos. La magia del mito y la leyenda  
175 hecho realidad en nuestro tiempo.

176 Uno escribe el hechizo correcto en el teclado y una  
177 pantalla cobra vida, mostrando cosas que nunca fueron  
178 ni podrían ser.

179

180 La programación es entonces divertida porque ella  
181 gratifica los anhelos contruidos en el interior de

182 nosotros y deleita las sensibilidades que nosotros  
183 tenemos en común con todos los hombres.

184

185

186 LOS MALES DEL ARTE.

187

188 No todo es placer, sin embargo, conociendo las  
189 dificultades inherentes se hace más fácil de llevar  
190 cuando ellas aparecen.

191

192 Primero, uno debe realizarse perfectamente. La  
193 computadora se asemeja a la magia de la leyenda a  
194 este respecto, también. Si un carácter, una pausa,  
195 del hechizo no está terminantemente en la forma  
196 apropiada, la magia no trabaja. Los seres humanos no  
197 están acostumbrados a ser perfectos, y pocas áreas de  
198 la actividad humana lo exigen. El ajuste al requisito  
199 para la perfección es, yo pienso, la parte más  
200 difícil de aprender a programar.

201

202 Luego, otra gente, ajusta unos objetivos, provee unos  
203 recursos y proporciona unas informaciones. Uno  
204 raramente controla las circunstancias de su trabajo o  
205 incluso su objetivo. En términos de gestión, la  
206 autoridad no es suficiente para su responsabilidad.  
207 Parece que en todos los ámbitos, sin embargo, los  
208 puestos de trabajo donde las cosas se han hecho nunca  
209 la autoridad oficial tiene consonancia con la  
210 responsabilidad. En la práctica, real (en  
211 contraposición a la formal) la autoridad se adquiere  
212 desde el momento de la realización.

213

214 La dependencia a los demás tiene un caso particular  
215 que es especialmente doloroso para el programador de  
216 sistemas. El depende de programas de otras personas.  
217 Estos son frecuentemente mal diseñadas, pobremente  
218 implementadas, entregadas incompletas (sin código  
219 fuente ni casos de estudio, y pobremente  
220 documentadas. Por lo que deben pasar horas estudiando

221 y reparando cosas que en un mundo ideal deberían  
222 estar completas, disponibles y utilizables.

223

224 El siguiente "ay" es que el diseño de grandes  
225 conceptos es divertido; encontrando pequeños errores  
226 no es más que trabajo. Con cualquier actividad  
227 creativa vienen monótonas horas de tedio, arduo  
228 trabajo, y la programación no es excepción

229

230 A continuación, se observa que la depuración tiene  
231 una convergencia lineal, o peor aún, cuando uno  
232 espera que de alguna manera una especie de enfoque  
233 cuadrático hasta el final. Así que la prueba se  
234 prolonga, el último error es mas difícil de encontrar  
235 que el primero.

236

237 El ultimo "ay", y muchas veces el ultimo "pitillo",  
238 es que el producto sobre el cual uno ha trabajado  
239 mucho tiempo, se hace obsoleto aun antes de haberlo  
240 terminado. Ya los colegas y los competidores están en  
241 la persecución de nuevas y mejores ideas. Ya el  
242 desplazamiento de el pensamiento de un niño no sólo  
243 es concebida, pero agendada.

244

245 Esto siempre luce peor de lo que realmente es. El  
246 nuevo y mejor producto es aquel generalmente no  
247 disponible cuando uno completa el propio; de él  
248 únicamente se ha hablado. El, también requiere de  
249 meses de desarrollo. El verdadero tigre nunca se  
250 empareja con el papel de uno a menos que se requiera  
251 su uso actual. A continuación, las virtudes de la  
252 realidad tienen una satisfacción de todos los suyos.

253

254 Pospuesto que la base tecnológica sobre la que uno  
255 construye está siempre avanzando. Tan pronto como uno  
256 congela un diseño, se convierte en obsoleto en  
257 termino de sus conceptos. Pero la implementación de  
258 productos reales requieren ajustes y cuantificación  
259 La obsolescencia de una implementación debe ser

260 medida en función de otras implementaciones, no en  
261 contra de conceptos no realizados.

262

263 El reto y la misión es encontrar soluciones reales  
264 para problemas reales, en el tiempo estimado, con los  
265 recursos disponibles.

266

267 Esta es entonces la programación, ambos un pozo de  
268 alquitrán en el que muchos esfuerzos han fracasado y  
269 una actividad creativa con alegrías y males propios.

270

271 Para muchos, las alegrías son muy superiores a los  
272 males, y para ellos el resto de este libro intentara  
273 establecer algunas tablas como puente sobre los pozos  
274 de alquitrán.

275

276 Juanca feb 22 de 2009.